An Efficient Graph Embedding Algorithm for a Three-Dimensional Cellular Reconfigurable Array

LOMTONG PAISAN*, MASATERU HARAO** and SHOICHI NOGUCHI*

The 3-dimensional cellular reconfigurable array(abbreviated to 3-D CECOA) is a reconfigurable multiprocessing system consisting of two uniformly interconnected network of cells, one of them is a rectangular array of processing elements called the active cells (AT-cells) and the other is a 3-dimensional lattice array of switch cells (SW-cells). This paper proposes a graph embedding algorithm, based on an edge-coloring technique for bipartite graphs, which is concerned with the embedding of an arbitrary graph into 3-D CECOA, where edges of a graph are mapped to paths in the array of SW-cells and nodes of a graph are mapped to AT-cells. The proposed algorithm takes $O(dn.\log(dn))$ time and $O(dn)$ space and requires $O((dn)^{3/2})$ volume of SW-cells, where d and n are the degree and the order of the given graph.

A distributed switch setting algorithm for establishing interconnection paths is called self-routing if each SW-cell determines its own setting depending on the incoming routing data. This paper also proposes a self-routing algorithm. Under this proposed self-routing algorithm, any interconnection path between AT-cells can be established using $4(\log((dn)^{1/2})+3)$ bits of the routing data, where d and n are the degree and the order of the given graph.

1. Introduction

Current integrated circuit technology is making feasible computer systems consisting of a large number of processing modules. In order to establish some highly efficient computing systems it is necessary to design multiprocessing systems which are able to compute problems utilizing their maximal parallelism. One such expective computer architecture will be a reconfigurable multiprocessing system [10, 11] which computes by configuring a processor network [13] according to the given problem. One of the most important but most difficult problems in the design of a reconfigurable multiprocessing system is the selection of an interconnection network. The performance of MIMD type multiprocessor systems depends mostly on the efficiency of their interconnection networks. Interconnection networks need to possess properties, such as universality concerning realizing any interconnection patterns, for suitability for VLSI implementation. Several reconfigurable interconnection networks are studied in [9]. It is also pointed out that a more flexible interconnection network is required to establish a dataflow computing system capable of executing a program using the maximal parallelism inherent to the problem [11].

In this paper, we shall propose a 3-dmensional cellular interconnection network(3-D CIN) as a reconfigurable interconnection network which is universal concerning realizing any interconnection paths among processors and being suitable for VLSI implementation, and a reconfigurable multiprocessing system called a 3-dimensional cellular reconfigurable array(3-D CECOA) having a 3-D CIN as its interconnection network.

Here, we shall discuss mainly the graph embedding algorithm which is universal concerning realizing any given graph into 3-D CECOA. The graph embedding algorithm is distributed into two algorithms. One of them is the set-up algorithm which computes the data for routing and the other is the routing algorithm which performs the switch setting according to the routing data.

In Section 2 we describe the formal definitions of this model and some related concepts. In Section 3 we give a more efficient algorithm based on the edge coloring of bipartite graphs which assures us we can realize an arbitrary graph into 3-D CECOA with reasonable cell volume. The design of routing patterns, routing data and routing algorithm are discussed in Section 4. Finally, Section 5 concludes the paper.

2. Formal Framework

2.1 3-D CECOA system

A three dimensional cellular reconfigurable array(3-D CECOA) system is a parallelepiped regular array of

*Research Institute of Electrical Communication, Tohoku University, Sendai, 980 Japan.
**Faculty of Engineering, Yamagata University, Yonezawa, 992 Japan.

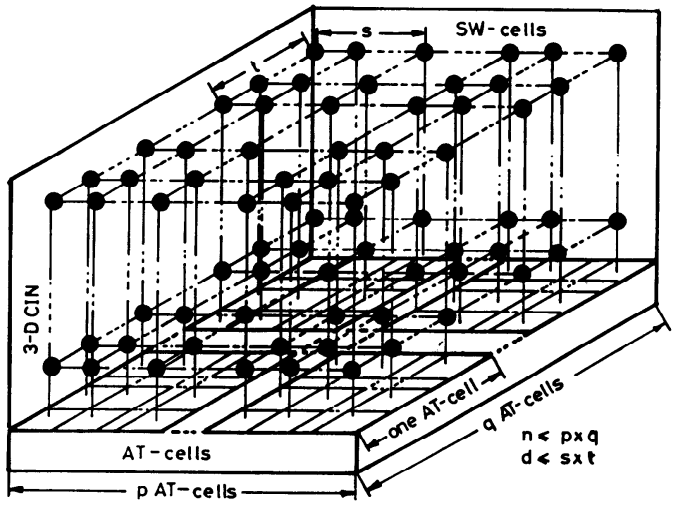

Fig. 1 3-D CECOA Systems.

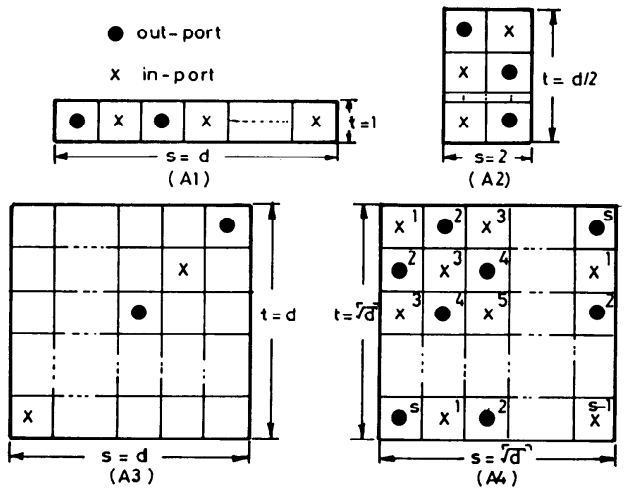

Fig. 2 Several examples of AT-cell organization.

two kinds of cells, one of which is the processing cell called the active cell(AT-cell) located at the bottom, and the other is the switch cell(SW-cell). The three dimensional lattice array of SW-cells is called the 3-dimensional cellular interconnection network(3-D CIN). By setting the switches of each SW-cell of the CIN corresponding to any given graphs we can realize connection topologies among the AT-cells which are homeomorphic to them. The rough sketch of a 3-D CECOA system is illustrated in Fig. 1, the 3-D CECOA system consisting of $p \times q$ AT-cells is denoted by $C_3(p \times q)$.

2.1.1 Organization of AT-cells

We assume that the degree d of the graphs to be realized in 3-D CECOA is arbitrary. When the degree of a given graph is d, each AT-cell of the 3-D CECOA has at least d ports. There are several different AT-cell organizations and, as we will see later, the cell organization effects the required cell volume in the 3-D CIN. In order to reduce the required layers of a 3-D CIN, *i.e.*, the size of the 3-D CIN, we distinguish the ports of each AT-cell to out-ports and in-ports. However, this restriction is not essential because bilateral communication is possible for the 3-D CIN by altering the mechanism of the cells moderately. Several examples of AT-cells for degree d graphs are illustrated in Fig. 2, where ● and × denote the out-ports and in-ports, respectively.

2.1.2 3-D CIN

A 3-D CIN is a network which is composed of identical SW-cells arrayed in a three dimensional lattice structure. Each SW-cell has six ports and each port is directly connected to a neighboring cell. A 3-D CIN consists of O(m) layers and each layer consists of $ps \times tq$ SW-cells, where $m = \max(ps, qt)/2$. The port of a SW-cell at the first layer which is connected to the AT-cell is called a terminal of the 3-D CIN(Fig. 1).

2.2 Basic definitions on graphs

An undirected graph(or simply graph) G is a pair (V, E) where V is the set of nodes, and $E = \{(u, v); u, v \in V\}$ is the set of edges. If E is replaced with an ordered pair $A = \{(u, v); u, v \in V\}$ then $G = (V, A)$ is called a directed graph(or simply digraph). Any such pair (u, v) is called an arc. Considering the graph as a d-way graph, if it is necessary to distinguish the edges(or arcs) adjacent with a node v, then we represent them such as (v, i), (v, j) etc. A walk of a graph G is an alternating sequence of nodes and edges $v_0, e_0, v_1, e_1, v_2, \ldots, v_{n-1}, e_n, v_n$ beginning and ending with nodes, in which each edge is incident with the two nodes immediately preceding and following it. This may also be denoted $v_0, v_1, \ldots, v_n$. It is a trail if all the edges are distinct, and a path if all the nodes are distinct. A walk is called eulerian if it traverses each edge once, goes through all nodes and ends at the starting node. An euler partition is a partition of the edges of a graph into open and closed paths, so each node of odd degree is the end of exactly one open path, and each vertex of even degree is the end of no open paths. A bipartite graph(or simply bigraph) is a graph whose node set V can be partitioned into two disjoint subsets V_1 and V_2 such that every edge of G joins V_1 with V_2. If more than one edge can join two nodes(multiple edges), then it is called a multigraph. For more detailed terminology refer to [5]. The graph of the k-dimensional cellular array is denoted $C_k = (C_k^l, S_k^l)$, where C_k^l is the k-fold Cartesian product of integer set I, *i.e.*, I^k and $S_k^l = \{(v, v+s_i); v \in V, s_i = (0, \ldots, 1^i, 0, \ldots, 0), i = 1, \ldots, k\}$. In this paper we only deal with the case $k = 3$.

2.3 Basic concepts on graph embedding

To realize any computation problems described by high level languages or dataflow languages into a 3-D CECOA, the computation problems are transformed into dataflow graphs and then the transformed dataflow graphs are embedded into the 3-D CECOA [12]. To embed any dataflow graphs into a 3-D CECOA, the graph embedding is considered to be a mapping function $f = (f_1, f_2)$ where f_1 maps the nodes of the graphs into AT-cells and f_2 maps the edges of the graphs into a 3-D CIN as interconnection paths among AT-cells satisfying the following conditions:

1) Duplicated use of the same port is not allowed.
2) Overlappings among interconnection paths are not allowed.
3) Crossings among interconnection lines are not allowed.

Interconnection paths in a 3-D CIN are considered as one-to-one mappings of the terminals of a 3-D CIN

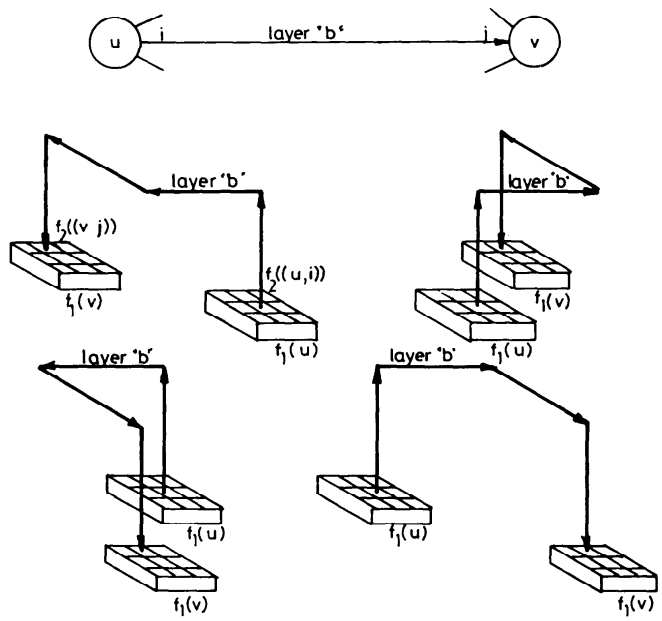

Fig. 3 Routing patterns.

under the restriction of the AT-cells assignment (Fig. 3).

From VLSI algorithm viewpoints the graph embedding problem can be taken as the hardware algorithm realization on chips, considering the graphs to be circuits. It has already been shown that any graph of degree six can be embedded in a three-dimensional cellular array using $O(n^{3/2})$ cell [4, 7, 8], where n is the order of graphs. Therefore the 3-D CIN has to be constructed using $cn^{3/2}$ SW-cells though the constant c should be taken as small as possible. It should be noted that there exist some differences between the VLSI algorithm realization problem and the interconnection pattern realization problem in 3-D CIN.

In the VLSI algorithm realization problem the given algorithm has to be embedded optimally under the area time estimation and the efficiency of the embedding is not important because the constructed VLSI algorithms are fixed. On the other hand, in the interconnection patterns realization problem, the main point is to design an efficient realization algorithm because the interconnection patterns will be reconfigured very often and the size of the 3-D CIN must be designed as small as possible taking into account the worst case.

Definition 1. An embedding $f=(f_1, f_2)$ of $G=(V, E)$ in $C_3=(C_3', S_3')$, $i=1, 2, 3$, is a mapping defined as follows:

1) f_1 is an injection from $V\times\{1, \ldots, d\}$ into $C_3'\times\{1, \ldots, p\}$ called the cell-port assignment, where d and p are the degree of G and the number of ports of an AT-cell respectively,
2) f_2 maps E into the paths of C_3 such that every pair of such paths is edge disjoint.

The graph embedding algorithm is divided into two algorithms, one of them is the set-up algorithm which determines a mapping f_1 and the other is the routing algorithm which establishes the interconnection paths in 3-D CIN using the data obtained by the set-up algorithm, *i.e.*, the construction of f_2.

3. Set-Up Algorithm

The set-up algorithm consists of two procedures, *i.e.*, the cell-port assignment and the layer assignment procedures.

3.1 Cell-port assignment procedure.

The cell-port assignment procedure can be described as follows:

1) Transformation to digraph: This procedure transforms a given graph to a digraph by assigning directions to each edge.
2) Cell assignment: This procedure defines the mapping ω_1 from node set V to the AT-cell set.
3) Port assignment: This procedure defines the mapping ω_2 from $V\times\{1, \ldots, d\}$ to $\omega_1(V)\times\{1, \ldots, s\cdot t\}$.

Proposition 1. Let G be an arbitrary graph of degree d and of order n. Then G can be transformed to a digraph G^* such that if d is even, then the outdegree and the indegree of each node are both $d/2$, and if d is odd, then one of the degrees of each node is $(d+1)/2$ and the other is $(d-1)/2$. The algorithm takes $O(dn)$ time and $O(dn)$ space.

Proof: There are an even number of nodes of odd degree. Accordingly we can transform G to a graph G_1 containing only even degree nodes by adding some edges to odd nodes. It is easy to see that this takes $O(n)$ time steps.

Since G_1 is eulerian [5], we find an eulerian trail. It is well known that the algorithm to find an eulerian trail take $O(e)$ time steps [1]. Next we assign each edge of G_1 a direction according to the direction of its eulerian trail. Let the resulting graph be $G_2=(V, A)$. From G_2, we delete the arcs which were added in the first stage of this procedure. Let the resulting graph be $G^*=(V, A^*)$. Clearly $G^*=(V, A^*)$ satisfies the condition of Proposition 1. @

Cell assignment is important to establish a compact CIN. Intuitively, the nodes adjacent with each other should be embedded into neighboring cells, but such a property depends on the characteristics of the given graphs. Therefore we don't here specify this procedure precisely, but the best cell assignments have been devised for trees, cubes and matrices [4], etc. The formal definition of cell-port assignment $f_1=(\omega_1, \omega_2)$ is given as follows:

a) ω_1 is an injection from V into C_3' where $C_3'=\{(x, y, 0); 0\leqslant x\leqslant p, 0\leqslant y\leqslant q\}$, and n is the order of V and $n=p\times q$.
b) ω_2 is an injection from $V\times\{1, \ldots, d\}$ into $C_3'\times\{1, \ldots, s\cdot t\}$ such that
   i) $\omega_2((v, i))$ is an out-port of $\omega_1(v)$ if (v, i) is an out-arc of v,
   ii) $\omega_2((v, i))$ is an in-port of $\omega_1(v)$ if (v, i) is an in-arc of v.

The cell-port assignment mapping f_1 is the composition of ω_1 and ω_2, *i.e.*, (ω_1, ω_2) defined as follows:

$$(\omega_1, \omega_2)((v, i))=(\omega_1(v), \omega_2(v, i))$$

for any

$$(v, i)\in V\times\{1, \ldots, d\}.$$

3.2 Layer assignment

In order to determine f_2, we use the layer assignment procedure. The layer assignment procedure can be described as follows:

1) Construct a bipartite graph G_B of order $sp+tq$ and of degree at most $D=\max(\lceil S/2\rceil .p, \lceil t/2\rceil .q)$,
2) Edge color G_B,
3) For a color i, $1\leqslant i\leqslant D$, construct the routing data of the interconnection paths using the routing patterns illustrated in Fig. 3, where $n\leqslant p\times q$ and $d\leqslant s\times t$.

3.2.1 Bipartite graph representation

It has been shown that the lower bound for volume to embed any permutation of n-inputs in 3-D VLSI is $\Omega((n)^{3/2})$ [8]. We shall show in this section an algorithm which realize any graph of degree d and of order n in a 3-D CIN with $O((dn)^{3/2})$ cells though its time complexity is polynomial. We assume here that the 3-D CECOA consists of $p\times q$ AT-cells and each AT-cell possesses $s\times t$ ports. By using only our proposed four connection patterns (Fig. 3), we can transform any mapped graph G in 3-D CECOA to a bipartite graph G_B. The procedure to construct a bipartite graph G_B can be described as follows:

Procedure: Bipartite graph construction
Input; A graph $G=(V, E)$;
A cell-port assignment $f_i=(\omega_1, \omega_2)$.
begin:

(1) Partition each AT-cell into $s\times t$ smaller segments, and associate the points r_j and c_i to the j-th row and the i-th column, where $1\leqslant j\leqslant t\times q$, $1\leqslant i\leqslant s\times p$.
(This is illustrated in Fig. 4)
(2) From G we derive a directed graph $G^*=(V, A)$.
(3) Define the (multi) graph $G_B=(X, B)$ such that
(i) $X=\{\{r_j\}\cup\{c_i\};\ 1\leqslant j\leqslant t\times q,\ 1\leqslant i\leqslant s\times p\}$.
(ii) $B=\{\{r_j, c_i\};\ f(u, a)\in r_j, f(v, b)\in c_i,$
$\{(u, a), (v, b)\}\in A\}$,
where a, b are the labels of this edge.
end.

Proposition 2. The graph $G_B=(W, B)$ is a bipartite graph of order $sp+tq$ and of degree at most $\max(\lceil s/2\rceil .p, \lceil t/2\rceil .q)$. (The size of G_B is the same as G)

Proof: From the construction of G_B, the edges exist only from row node to column node. Hence G_B is a bipartite graph. The order is equal to the total number of row nodes and column nodes. Since the number of out-port (in-port) of each row and of each column of an AT-cell are at most $\lceil s/2\rceil$ and $\lceil t/2\rceil$ respectively, the total number of edges connected to out-ports (in-ports) of each row and column of the first layer are at most $\lceil s/2\rceil .p$ and $\lceil t/2\rceil .q$ respectively. @

3.2.2 Layer assignment by edge-coloring graph

An edge-coloring of a graph G is an assignment of colors to its edges so that no two adjacent edges are assigned the same color. Generally the problem of edge-coloring belongs to the class of NP-complete problems, but for the class of bipartite graphs more efficient algorithms are available.

We use the modified algorithm of color-by-partition [3]. It is to be noted that the minimum number of required colors is equal to the degree of the bipartite graph.

Lemma 1. Modified color-by-partition algorithm finds a minimum coloring, in time $O(|V|^2 \log |V|)$ and space $O(|E|+|V|)$, where $|V|$ and $|E|$ denote the order and size of G_B respectively.

Procedure: color-by-partition [3].
comment: G is a bipartite graph, all of whose edges are uncolored. A minimum coloring of G is found.
begin:
let D be the maximum degree in G;
if $D=1$ then color all edges in G using a new color,
else begin;
divide G into edge-disjoint subgraphs G_1 and G_2 having maximum degree D_1 and D_2, where D_1, $D_2\leqslant\lceil D/2\rceil$ and G_1 has no more edges than G (euler partition [2]);
color-by-partition (G_1);
remove the edges of r colors from G_1 and add them to G_2,
where $r=\lceil 2^{\log(d/2)}\rceil - D_2$;
euler color [2] (G_2); comment: now the coloring of G_1 and G_2 gives a D-coloring

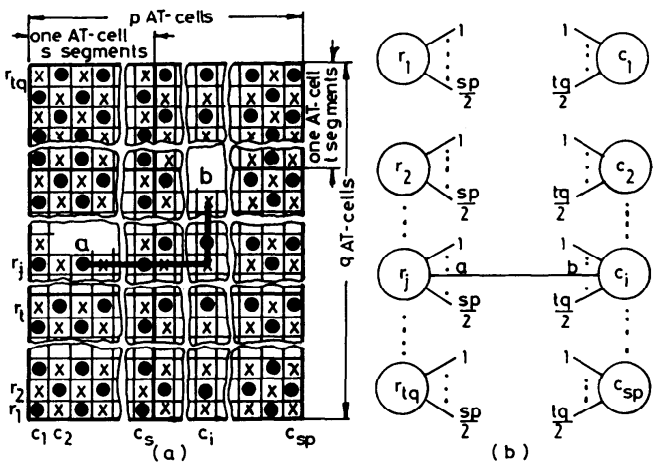

Fig. 4 Bipartite graph representation of G.

Table 1 AT-cell organization and required SW-cells.

	Cell types			
	A1	A2	A3	A4
Colors(layers)	$\frac{b}{2}m$	$\frac{d}{4}m$	m	$\frac{\lceil\sqrt{d}\rceil}{2}m$
Area of one AT-cell	d	d	d^2	d
Area of one layer	dm^2	dm^2	d^2m^2	dm^2
Required SW-cells	$\frac{d^2}{2}m^3$	$\frac{d^2}{4}m^3$	d^2m^3	$\frac{(\sqrt{d}\,m)^3}{2}$

$m=p=q=\lceil\sqrt{n}\rceil$.

or $D+1$-coloring of G;
if G is not D-colored
then begin;
make all edges of some color β uncolored;
for all uncolored edges e do augment [3] (e);
end.
end.
end color-by-partition.

We assign the edge colored by "b" to the b-th layer of the 3-D CIN.

Since the usable routing patterns are restricted to the ones of Fig. 3, no overlapping arises among interconnection paths. This completes the construction of the mapping f_2.

Lemma 2. Let G_B by any bigraph. Then G_B can be realized in 3-D CIN without overlapping using the 4 routing patterns in Fig. 3.

Theorem 1. An arbitrary graph G of degree d and of order n can be realized in 3-D CECOA using at most $D=\max(\lceil s/2\rceil .p,\ \lceil t/2\rceil .q)$ layers.

Proof: For G, the bipartite graph G_B is of degree $D=\max(\lceil s/2\rceil .p,\ \lceil t/2\rceil .q)$ and of order $V=s.p+t.q$ by Proposition 2. Hence G_B can be D colorable. This implies that G_B can be embedded using D layers. @

Example 1: The degree of derived bipartite graphs corresponding to the AT-cells (A1), (A2), (A3) and (A4) of Fig. 2 are given as $(dm)/2$, $(dm)/4$, m and $(d^{1/2}m)/2$, respectively. The required volume of 3-D CIN, etc., are also given in Table 1.

3.3 AT-cell organization for compact 3-D CIN

From the results of previous section, the number of required layers depends on the organization of the AT-cell. In order to reduce the number of required layers, we have to construct the bipartite graph of smallest possible degree. As we can observe in Table 1, an interesting AT-cell is the one of (A4) in Fig. 2, which is designed using the property of Latin squares.

A Latin square is a matrix having integers as its components of such that each distinct integer occurs exactly once in each row and each column of it. For designing an AT-cell of d-ports, we use a $s\times s$ Latin square, where $s=d^{\lceil 1/2\rceil}$.

Roughly speaking we assign the ports of even number to either the out-arcs or the in-arcs, and the odd ones to the remaining arcs. If the number of out-arcs(or in-arcs) of certain nodes exceeds $d/2$, then an extra port not used is to be assigned further (Fig. 5). It is noted that the number of out-ports(or in-ports) in each column or each row of the matrix is at most $s/2$, where $s=d^{\lceil 1/2\rceil}$.

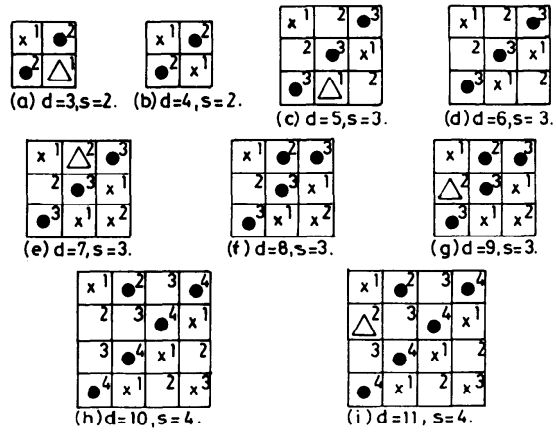

Fig. 5 AT-cell organization based on Latin square.

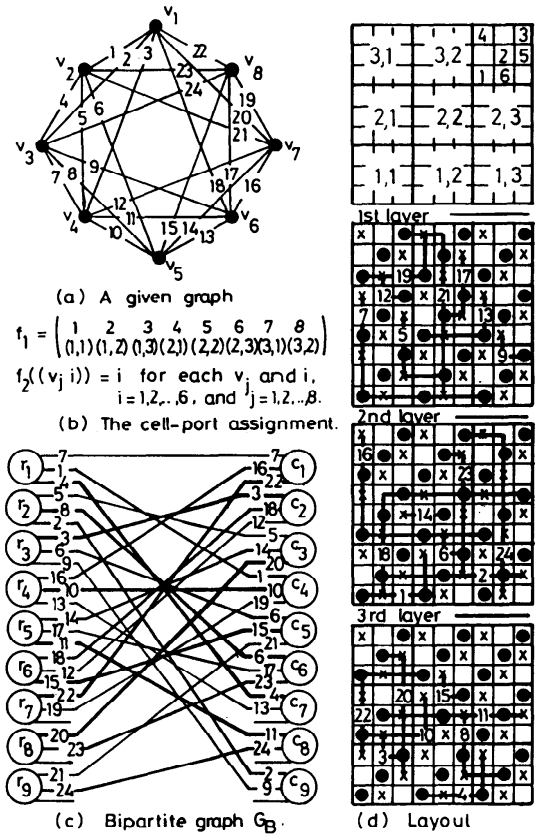

Fig. 6 For example 2.

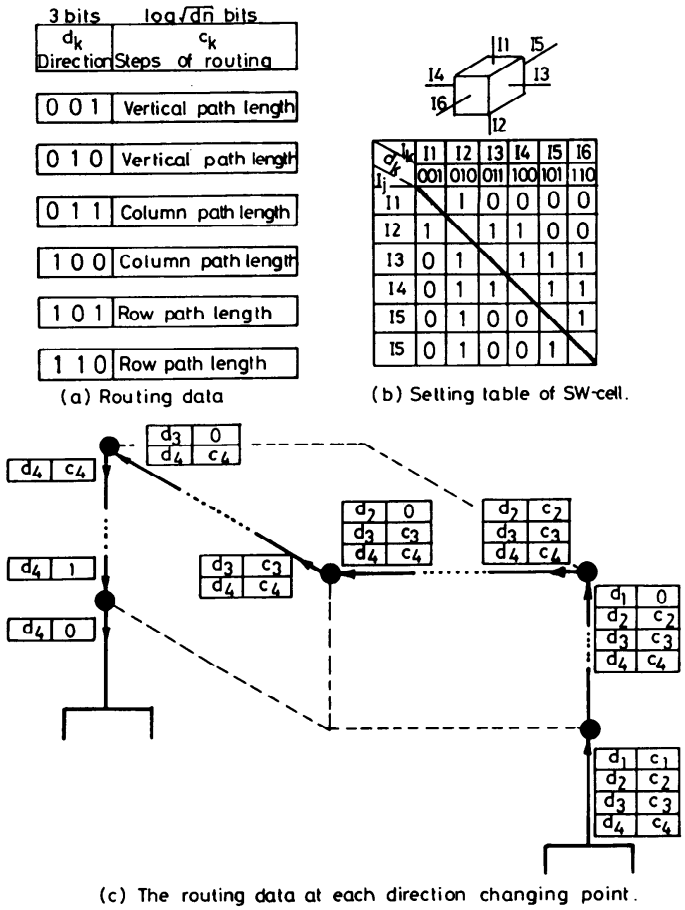

Fig. 7 Routing algorithm.

Theorem 2. (Main theorem) An arbitrary graph G of degree d and of order n can be realized in CIN with $D=(1/2)(dn)^{1/2}$ layers and therefore with $(1/2)(dn)^{3/2}$ SW-cells. The realization algorithm runs in $O(dn.\log(dn))$ time and $O(dn)$ area.

Proof: From the construction of standard AT-cell we have

$$s=t=\lceil d^{1/2}\rceil,\ p=q=\lceil n^{1/2}\rceil.$$

Hence the order, the size and the degree of the derived bipartite graph are given as follows:

$$|V|=2(dn)^{1/2},\ |E|=(dn)/2 \text{ and } |D|=(1/2)(dn)^{1/2}.$$

By applying these relations to Theorem 1 and Lemma 1, we obtain the results. @

Example 2: We embed the graph given in Fig. 6(a). First we define the cell-port assignment $f_1=(\omega_1, \omega_2)$ as given in Fig. 6(b). Under this mapping f_1, the bigraph G_B is obtained as shown in Fig. 6(c). Thus we have the interconnection of Fig. 6(d).

4. Routing Algorithm

4.1 Routing patterns

In general there are a number of routing patterns to realize an arc of the graphs in a 3-D CIN. If we use some complex routing patterns, then complex routing data may be required. It is also noted that the volume of the required SW-cell is $O((dn)^{3/2})$ even if some sophisticated routing patterns are used, where d and n are the degree and the order of the given graph. From these reasons, we adopt the routing patterns illustrated in Fig. 3.

4.2 Routing control mechanism

For routing control there are several methods such as packet switching and circuit switching though the described method is available for both cases. We show the control mechanism for routing of the SW-cells in Fig. 7(b), where I_j connects with I_k under the input d_k iff the component (I_j, I_k) is 1. We denote it as $\delta(I_j, d_k)=I_k$.

4.3 Routing data

Since we use only the simple routing patterns illustrated in Fig. 3, there are at most three direction changing points of the routing patterns in 3-D CIN. For the routing data, we have only to indicate the direction changing point. For the SW-cells of a 3-D CIN, one data of three bits is required to indicate the direction of the routing respectively. If we use one data for one step of routing, then the required routing data for the 3-D CIN is $O((dn)^{1/2})$ bits. To minimize the length of the routing data, we apply the step count down procedure. The step-count-down procedure can be described as follows.

1) Code the total routing path length of each routing direction into the binary representation. It requires $\log(m)$ bits of data for this if the length of the routing direction is m. Obviously, $m \leqslant (dn)^{1/2}$. Therefore, each routing datum is in the form of $d_k \times c_k$, where d_k is the routing direction information indicating the routing direction and c_k is the routing path length information indicating the number of necessary SW-cells to realize the given routing path (Fig. 7(a)). The routing data consists of four such data.
2) When a routing data passes a SW-cell, the data indicating the routing path length are decreased by one.

4.4 Self-routing Algorithm

A distributed switch setting mechanism for routing is called self-routing if each switch cell determines its own setting depending only on the preset incoming routing data [6]. Now we shall present the self-routing algorithm for each SW-cell of a 3-D CIN. The self-routing algorithm can be described as follows:

Procedure: Self-routing (At each SW-cell)

Input: Routing data $D_k D_{k+1}, \ldots, D_r$(assume that the routing data are received at a port I_j), where $D_a = d_a \times c_a$ and $k \leqslant a \leqslant r$.

begin:

a) Get the data c_k of D_k.
   If $c_k \neq 0$
   then D_k is update to $d_k \times (c_k - 1)$ and connect I_j to $\delta(I_j, d_k)$.
   else Delete D_k and get d_{k+1} of D_{k+1}.
   Connect I_j to $\delta(I_j, d_{k+1})$.
b) Send the routing data to the next SW-cell.

end;

The routing data at each direction changing point of the routing paths in the 3-D CIN is illustrated in Fig. 7(c). Noted that in order to connect I_j with I_k, the input is always d_k independent of the input port I_j. By this property we can show the routing procedure works correctly.

Theorem 3. The routing procedure works correctly using routing data of length $4(\lceil \log(m^{1/2}) \rceil + 3)$ bits if we use the routing patterns of Fig. 3, where m is the total number of ports of the AT-cells, that is, if each AT-cell possesses d-ports and n AT-cells exist, then m is dn.

Proof: We assume a situation such that the routing data $D_i D_{i+1}, \ldots, D_k$ are routed to a port I_j.

Case 1: If $D_i = d_i \times c_i$ and $c_i \geqslant 1$, then D_i is updated to $d_i \times (c_i - 1)$, if d_i is "d_k", then I_j is connected to I_k by D_i according to the above discussion.

Case 2: If $D_i = d_i \times c_i$ and $c_i = 0$, then D_i is deleted and the next data $D_{i1} = d_{i+1} \times c_{i+1}$ will be read. No matter what the present port I_j may be, if d_{i+1} is "d_k", then I_j is connected to I_k by D_{i+1} according to the above discussion.

This realizes the correct direction changing. By using the step-count-down technique, the required routing data for each direction routing is $3 + \lceil \log(m^{1/2}) \rceil$ and there are at most four routing directions in 3-D CIN. Therefore, the required routing data is $4(\lceil \log(m^{1/2}) \rceil + 3)$. This concludes Theorem 3. @

Theorem 4. No deadlock occurs for the operation of the self-routing procedure in the SW-cells of a 3-D CIN.

Proof: From the set-up algorithm, the case such that two or more connection paths are embedded on the same row or column path never occurs. Thus, no two or more routing data are routed through the same input and/or the same output port at the same time.

Therefore, no deadlock occurs for the operation of the self-routing procedure of each SW-cell of the 3-D CIN. @

5. Conclusion

In this paper we mainly discussed the theoretical aspects of 3-D CIN as an interconnection network for the reconfigurable multiprocessing system 3-D CECOA and it has been shown that the proposed 3-D CIN possesses sufficient ability for such purpose. One of the advantages of 3-D CIN is the possibility of parallel switch settings. To establish a parallel, asynchronous and complete distributive switch setting algorithm, a more detailed SW-cell organization or communication mechanism of a 3-D CIN must be studied. We implemented also a layout system which outputs the interconnection patterns among AT-cells from the given graph G and the cell-port assignment function. This can be seen to be a simulator of the interconnection network and it is ensured that the algorithm works quite reasonably.

In order to construct a 3-D CECOA, it is also necessary to study more precisely the I/O control mechanism between host computer and CECOA system, the total performance estimation etc. One of the other attractive related research areas is the application to hardware algorithm design based on a programmable model. Several interesting algorithms such as a pattern matching machine, FFT, polynomial evaluation etc., have also been discussed using this approach [12].

Acknowledgements

The authors would like to thank for the refree for his careful reading and helpful comments. This work was partially supported by the national scientific research grant-in-aid, general research (c) 60580016.

References

1. AHO, A. V., HOPCROFT, J. E. and ULLMAN, J. D. Design and Analysis of Computer Algorithm, Addison-Wesley (1974).
2. GABOW, H. N. Using Euler Partitions to Edge Color Bipartite Multigraphs, *Int. J. Comput. and Inf. Science*, Vol. 14 (1976) 345–355.
3. GABOW, H. N. and KARIV, O. Algorithm for Edge-coloring Bipartite Graph and Multigraphs, *SIAM J. Comput.* Vol. 11 (1982) 117–129.
4. HARAO, M., LOMTONG, P. and NOGUCHI, S. Graph Realization Algorithms for Interconnection Networks of Reconfigurable Parallel Processing System, *IECE, Tech. Commit. of Automaton and Language*, AL 83–10 (1983).
5. HARARY, F. Graph Theory, Addison-Wesley (1969).
6. NASSIMI, D. and SAHNI, S. A Self-routing Benes Network, *IEEE Trans. Comput.* Vol. C-30 (1981) 332–340.
7. PREPARATA, F. P. *Optimal Three-Dimensional VLSI Layouts, Math. Sys. Theory*, Vol. 16 (1983).
8. ROSENBERG, A. L. Three Dimensional VLSI, A Case Study, *IBM T. J. Watson Res.*, RC 8745 (1981).
9. SIEGEL, H. J. Interconnection Networks for Parallel and Distributed Processing, *IEEE Trans. Comput.* Vol. C-30, No. 4 (1981).
10. SNYDER, L. Introduction to a Configurable Highly Parallel Computer, *IEEE Computer* (1982) 47–56.
11. DENNIS, J. B. Building Blocks for Dataflow Prototypes, *in Proc. Symp. Comput. Arch.* (1980).
12. LOMTONG, P., HARAO, M. and NOGUCHI, S. Hardware Algorithm Realization with Cellular Reconfigurable Arrays, *IECE. Tech. Commit. of Automaton and Language*, AL83-28 (1983).
13. ULLMAN, J. D. Computational Aspects of VLSI, *Computer press* (1984).

(Received April 4, 1985; revised August 12, 1985)